PCT/AU2004/001087

Patent Office
Canberra

I, LEANNE MYNOTT, MANAGER EXAMINATION SUPPORT AND SALES hereby certify that annexed is a true copy of the Provisional specification in connection with Application No. 2003904351 for a patent by SILVERBROOK RESEARCH PTY LTD. as filed on 15 August 2003.

PRIORITY DOCUMENT
SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)

WITNESS my hand this
Twenty-fifth day of August 2004

LEANNE MYNOTT
MANAGER EXAMINATION SUPPORT AND SALES

SYSTEMS, METHODS AND APPARATUS

Jon Napper
jon.napper@silverbrookresearch.com

Silverbrook Research Pty Ltd
393 Darling Street, Balmain NSW Australia
15 August 2003

1 Introduction

Recent advances in pattern classification have enabled the development of sophisticated software systems that can recognize natural language input such as speech [6] or handwriting [7]. These applications allow users to communicate with the system in a natural and convenient way, and permit the automation of tasks that previously required human input. Some examples of such applications include interactive voice response (IVR) systems, automated cheque-processing systems, and automated form data-entry systems.

In addition, the growth of networked computing and the Internet has enabled the development of complex distributed systems, and the existence of open, standardized protocols has allowed the integration of end-user devices, centralized servers, and applications. An example of a three-tiered distributed system architecture is depicted in Figure 1. The combination of distributed computing and pattern recognition techniques has made possible the development of systems such as Netpage [1], an interactive paper-based interface to online information. Systems such as these give users the ability to interact with their information from any location that provides network connectivity (including wireless network access) using familiar human-communication techniques such as handwriting or speech.

This paper discusses some of the technical issues involved with integrating pattern recognition techniques into a distributed environment, and proposes a generic framework for performing distributed recognition using centralized recognition servers and distributed context processing. Also discussed are techniques for managing the user-specific customisation and adaptation that is required to make pattern recognition systems accurate and flexible.

1.1 Cross-References

Various methods, systems and apparatus relating to the present invention are disclosed in the following co-pending applications filed by the applicant or assignee of the present invention. The disclosures of all of these co-pending applications are incorporated herein by cross-reference.

5 October 2002:

Australian Provisional Application 2002952259, "Methods and Apparatus (NPT019)".

15 October 2002:

PCT/AU02/01391, PCT/AU02/01392, PCT/AU02/01393, PCT/AU02/01394 and PCT/AU02/01395.

26 November 2001:

PCT/AU01/01527, PCT/AU01/01528, PCT/AU01/01529, PCT/AU01/01530 and PCT/AU01/01531.

11 October 2001:

PCT/AU01/01274.

14 August 2001:

PCT/AU01/00996.

27 November 2000:

PCT/AU00/01442, PCT/AU00/01444, PCT/AU00/01446, PCT/AU00/01445, PCT/AU00/01450, PCT/AU00/01453, PCT/AU00/01448, PCT/AU00/01447, PCT/AU00/01459, PCT/AU00/01451, PCT/AU00/01454, PCT/AU00/01452, PCT/AU00/01443, PCT/AU00/01455, PCT/AU00/01456, PCT/AU00/01457, PCT/AU00/01458 and PCT/AU00/01449.

20 October 2000:

PCT/AU00/01273, PCT/AU00/01279, PCT/AU00/01288, PCT/AU00/01282, PCT/AU00/01276, PCT/AU00/01280, PCT/AU00/01274, PCT/AU00/01289, PCT/AU00/01275, PCT/AU00/01277, PCT/AU00/01286, PCT/AU00/01281, PCT/AU00/01278, PCT/AU00/01287, PCT/AU00/01285, PCT/AU00/01284 and PCT/AU00/01283.

15 September 2000:

PCT/AU00/01108, PCT/AU00/01110 and PCT/AU00/01111.

30 June 2000:

PCT/AU00/00762, PCT/AU00/00763, PCT/AU00/00761, PCT/AU00/00760, PCT/AU00/00759, PCT/AU00/00758, PCT/AU00/00764, PCT/AU00/00765, PCT/AU00/00766, PCT/AU00/00767, PCT/AU00/00768, PCT/AU00/00773, PCT/AU00/00774, PCT/AU00/00775, PCT/AU00/00776, PCT/AU00/00777, PCT/AU00/00770, PCT/AU00/00769, PCT/AU00/00771, PCT/AU00/00772, PCT/AU00/00754, PCT/AU00/00755, PCT/AU00/00756 and PCT/AU00/00757.

24 May 2000:

PCT/AU00/00518, PCT/AU00/00519, PCT/AU00/00520, PCT/AU00/00521, PCT/AU00/00522, PCT/AU00/00523, PCT/AU00/00524, PCT/AU00/00525, PCT/AU00/00526, PCT/AU00/00527, PCT/AU00/00528, PCT/AU00/00529, PCT/AU00/00530, PCT/AU00/00531, PCT/AU00/00532, PCT/AU00/00533, PCT/AU00/00534, PCT/AU00/00535, PCT/AU00/00536, PCT/AU00/00537, PCT/AU00/00538, PCT/AU00/00539, PCT/AU00/00540, PCT/AU00/00541, PCT/AU00/00542, PCT/AU00/00543, PCT/AU00/00544, PCT/AU00/00545, PCT/AU00/00547, PCT/AU00/00546, PCT/AU00/00554, PCT/AU00/00556, PCT/AU00/00557, PCT/AU00/00558, PCT/AU00/00559, PCT/AU00/00560, PCT/AU00/00561, PCT/AU00/00562, PCT/AU00/00563, PCT/AU00/00564, PCT/AU00/00565, PCT/AU00/00566, PCT/AU00/00567, PCT/AU00/00568, PCT/AU00/00569, PCT/AU00/00570, PCT/AU00/00571, PCT/AU00/00572, PCT/AU00/00573, PCT/AU00/00574, PCT/AU00/00575, PCT/AU00/00576, PCT/AU00/00577, PCT/AU00/00578, PCT/AU00/00579, PCT/AU00/00581, PCT/AU00/00580, PCT/AU00/00582, PCT/AU00/00587, PCT/AU00/00588, PCT/AU00/00589, PCT/AU00/00583, PCT/AU00/00593, PCT/AU00/00590, PCT/AU00/00591, PCT/AU00/00592, PCT/AU00/00594, PCT/AU00/00595, PCT/AU00/00596, PCT/AU00/00597, PCT/AU00/00598, PCT/AU00/00516, PCT/AU00/00517 and PCT/AU00/00511.

1.2 Pattern Recognition

The basic processing steps of a pattern recognition system are depicted in Figure 3. Processing begins when an input device generates a signal that is to be recognized by the system (that is, to be classified as belonging to a specific class or sequence of class elements). Usually, one or more pre-processing procedures are applied to remove noise and normalize the signal, which is then segmented to produce a stream of primitive elements required for the classification procedure. Note that often this segmentation is "soft", meaning that a number of potential segmentation points are located, and the final segmentation points are resolved during classification or context processing.

The segmented signal is then passed to a classifier where a representative set of features is extracted from the signal and used in combination with a pre-defined model of the input signal to produce a set of symbol hypotheses. These hypotheses give an indication of the probability that a sequence of segments within the signal represent a basic symbolic element (e.g. letter, word, phoneme, etc.). After classification, the context-processing module uses the symbol hypotheses generated by the classifier to decode the signal according to a specified context model (such as a dictionary or character grammar). The result produced by the context processing is passed to the application for interpretation and processing.

1.3 Context Processing

Natural language input is inconsistent, noisy, and ambiguous, leading to potential recognition and decoding errors. However, high recognition accuracy is required for pattern recognition applications to operate successfully, since mistakes can be expensive and frustrating to users. As a result, recognition systems must make use of as much contextual information as possible to increase the possibility of correctly recognizing the input. For example, when recognizing a signal that must represent a country name, the recognition system can use a pre-defined list of valid country names to guide the recognition procedure. Similarly, when recognizing a phone number, a limited symbol set (i.e. digits) can be used to constrain the recognition results.

The problem domain for many pattern recognition systems is inherently ambiguous (i.e. many of the input patterns encountered during processing cannot be accurately classified without further information from a different source). As an example, one of the major issues faced in the development of highly accurate handwriting recognition systems is the inherent ambiguity of handwriting (e.g. the letters 'u' and 'v', 't' and 'f', and 'g' and 'y' are often written with a very similar appearance and are thus easily confused). Human readers rely on contextual knowledge to correctly decode handwritten text, and as a result a large amount of research has been directed at applying syntactic and linguistic constraints to handwritten text recognition [8,9,10,11,12,13]. Similar work has been performed in the field of speech recognition, natural language processing, and machine translation.

1.4 Current Systems

Paragraph offers a network-based distributed handwriting recognition system called NetCalif [2] that is based on their Calligraphy handwriting recognition software. "The user's natural handwriting - cursive, print, or a combination of both - is captured by compact client software, then transmitted from the Internet-connected device to the NetCalif servers where it is converted and returned as typewritten text to the device" [2].

Philips offers SpeechMagic, "a client/server-based, professional speech recognition software package" [3]. This system supports specialized vocabularies (called ConTexts) and "dictation, recognition, and correction can be done, independently of the location, across a LAN, WAN, or the Internet" [3].

1.5 Detailed Description of the Preferred Embodiments

In the preferred embodiment, the invention is configured to work with the Netpage networked computer system, a detailed description of which is given in our co-pending applications, including in particular PCT application WO0242989 entitled "Sensing Device" filed 30 May 2002, PCT application WO0242894 entitled "Interactive Printer" filed 30 May 2002, PCT application WO0214075 "Interface Surface Printer Using Invisible Ink" filed 21 February 2002, PCT application WO0242950 "Apparatus For Interaction With A Network Computer System" filed 30 May 2002, and PCT application WO03034276 entitled "Digital Ink Database Searching Using Handwriting Feature Synthesis" filed 24 April 2003. It will be appreciated that not every implementation will necessarily embody all or even most of the specific details and extensions described in these applications in relation to the basic system. However, the system is described in its most complete form to assist in understanding the context in which the preferred embodiments and aspects of the present invention operate.

In brief summary, the preferred form of the Netpage system provides an interactive paper-based interface to online information by utilizing pages of invisibly coded paper and an optically imaging pen. Each page generated by the Netpage system is uniquely identified and stored on a network server, and all user interaction with the paper using the Netpage pen is captured, interpreted, and stored. Digital printing technology facilitates the on-demand printing of Netpage documents, allowing interactive applications to be developed. The Netpage printer, pen, and network infrastructure provide a paper-based alternative to traditional screen-based applications and online publishing services, and supports user-interface functionality such as hypertext navigation and form input.

Typically, a printer receives a document from a publisher or application provider via a broadband connection, which is printed with an invisible pattern of infrared tags that each encodes the location of the tag on the page and a unique page identifier. As a user writes on the page, the imaging pen decodes these tags and converts the motion of the pen into digital ink. The digital ink is transmitted over a wireless channel to a relay base station, and then sent to the network for processing and storage. The system uses a stored description of the page to interpret the digital ink, and performs the requested actions by interacting with an application.

Applications provide content to the user by publishing documents, and process the digital ink interactions submitted by the user. Typically, an application generates one or more interactive pages in response to user input, which are transmitted to the network to be stored, rendered, and finally printed as output to the user. The Netpage system allows sophisticated applications to be developed by providing services for document publishing, rendering, and delivery, authenticated transactions and secure payments, handwriting recognition and digital ink searching, and user validation using biometric techniques such as signature verification.

2 Distributed Pattern Recognition

An example architecture for a distributed pattern recognition system is depicted in Figure 2. In the example, a signal is recorded at the client and transmitted over a network to a server for recognition, with the results transmitted back to the client or a third party application for interpretation and processing. One advantage of this approach is that client devices and distributed application systems do not require the significant computing resources commonly needed to perform pattern recognition, and the network servers that perform the recognition are not subject to the resource constraints that are inherent in many client devices (e.g. mobile phones, personal-digital assistants, imaging pens, etc.). As a result, they are able to use extremely processor- and/or memory-intensive techniques to improve recognition accuracy, and can use hardware optimised to perform the specific recognition task.

Performing pattern recognition on a centralized server also offers an advantage to pattern-recognition systems that employ user-specific adaptation to achieve higher recognition rates. For example, some handwriting recognition techniques develop a handwriting model for each user of the system based on previous recognition results, which is then used to improve the future accuracy of the system for that user [4,5]. This adaptation is more effective if a single server performs all recognition for a user (rather than a large number of individual applications each performing their own recognition), since the server is able to perform adaptation based on the input generated by all applications. In addition to this, centralized server-based pattern recognition simplifies the management of the recognition system by allowing recognizers to be reconfigured and upgraded without interaction with the distributed clients and applications, and allows training and test data to be easily collected.

However, the information required to perform the context processing stage of a pattern recognition system is generally application specific and is often very large (e.g. entries in a large application-specific database), making it impractical to transmit the context information to a centralized server for processing. A solution to this problem is to use a mechanism for distributed recognition as depicted in Figure 2. When a user generates a signal to be recognized and processed by an application, the signal is submitted to a distributed server for processing. The server performs pre-processing, segmentation, and classification, but does not use a context model to decode the result (or only performs partial decoding as described in the following discussion). Rather, the intermediate recognition results are returned to the application allowing the application to apply any arbitrarily complex and domain-specific context processing to decode the signal.

2.1 Symbol DAG

One method of returning the intermediate recognition results to an application is to use a symbol DAG (directed acyclic graph), which is a generic data structure that contains symbol and associated scores as vertices, and valid transitions between symbols as edges. The structure can be implemented as a two-dimensional array of elements, each of which defines the output generated by the pattern classifier for a single segment of the signal and the associated valid transitions for that segment. This structure represents all the potential recognition alternatives that may be derived from the input signal based on the results of the classifier. The application uses this structure, in combination with a context model, to decode the input signal.

The symbol DAG is equivalent to a matrix where each column contains the results of the classification of a single segment of the input signal. Each element in the column represents the probability that the classified segment is a particular symbol, and includes an offset that indicates the next possible segment (column) in the input signal that can follow this symbol. Thus, the matrix represents all the possible decoding paths based on the output of the pattern classifier. These paths and associated classification scores can be combined with a context model to fully decode the input signal.

Note that the symbol DAG is applicable in any pattern recognition task where a sequence of classification results is decoded using a context or set of constraints. The symbols contained in the symbol DAG may be any primitive element that is generated as the output of a pattern classifier, including the output from a time-series classifier. Examples of such recognition systems include handwriting and speech recognition, protein sequencing [14], image processing and computer vision [15], and econometrics [16].

2.2 Symbol DAG Example

As an example, Table 1 shows a symbol DAG that represents the output from a handwritten character recognizer generated by the ambiguous text given in Figure 4. In this example, the recognizer has found two possible character segmentation arrangements, as depicted by the two rows in the symbol DAG. Note that in the examples, the symbol scores are given as probabilities; however, an actual implementation will typically use log-probabilities (i.e. the base-10 logarithm of the probability result) to improve the performance of context processing and to avoid overflow and underflow problems that occur when multiplying probabilities using finite precision floating-point operations.

To decode the alternatives, the context processor starts with the first entry in the DAG (i.e. the character 'c'). The score for this entry is added to the accumulated total (since log-probabilities are added rather than multiplied), and processing moves to the column given by the offset value in the entry (in this example, column 1). In column 1, two alternatives exist (i.e. "cl" or "cb"), and the scores for these alternatives are found by adding the scores to the previous total. The decoding continues until the end of the DAG is reached. Similarly, the second entry in column 0 (i.e. the character 'd') is decoded; note however, that column 1 is skipped in this traversal of the DAG, as indicated by the offset value of 2 in the character score entry. This is due to the letter 'd' being constructed using two strokes, and thus the recognition of the letters 'l' and 'b' cannot be valid in this alternative. Thus, the potential decoding alternatives in this example are:

```
clog  = 0.7 * 0.8 * 1.0 * 1.0 = 0.56
cbg   = 0.7 * 0.2 * 1.0       = 0.14
dog   = 0.3 * 1.0 * 1.0       = 0.30
```

These values can now be combined with a language model or other contextual information to select the most likely word.

	0	1	2	3
Character	c	l	o	g
Offset	1	2	3	0
Score	0.7	0.8	1.0	1.0
Character	d	b		
Offset	2	3		
Score	0.3	0.2		

Table 1. Example DAG for "clog"/"dog" Ambiguity

The DAG structure must ensure that strokes are assigned to an individual letter only once. To do this, alternate paths must be defined to ensure that if a stroke is assigned to a letter, no subsequent letter may use that stroke in its construction. An example of this is given in Figure 5, with the derived DAG depicted in Table 2. In this example, the short, horizontal marks can potentially be recognized as crossbar elements of a letter 't', or diacritical marks for the letter 'i'. However, if a marking is used as a crossbar, it cannot subsequently be used as a diacritical. The potential decoding alternatives in this example are:

```
tile = 0.6 * 1.0 * 0.6 * 1.0 = 0.36
tite = 0.6 * 1.0 * 1.0 * 1.0 = 0.60
```

```
lite = 0.4 * 1.0 * 1.0 * 1.0 = 0.40
```

These values can now be combined with a language model to select the most likely word.

	0	1	2	3	4	5
Character	t	i	i	t	l	e
Offset	1	4	3	5	5	-
Score	0.6	1.0	1.0	1.0	0.6	1.0
Character	l				t	
Offset	2				5	
Score	0.4				0.4	

Table 2. Example DAG for "lite"/"tile" Ambiguity

Additionally, the character value of a DAG entry can be set to zero, indicating a NUL character (i.e. a character that does not change the text, but will modify the text probability). This allows word break positions (i.e. spaces) to be modeled as a SPACE/NUL pair, indicating that there is a certain probability that a space appears at that point in the DAG. For example:

	0	1	2
Character	a	NUL	b
Offset	1	1	-
Score	1.0	0.6	1.0
Character		SPACE	
Offset		1	
Score		0.4	

Table 3. Example DAG for SPACE/NUL Pair

The potential decoding alternatives in this example are:

```
ab  = 1.0 * 0.6 * 1.0 = 0.6
a b = 1.0 * 0.4 * 1.0 = 0.4
```

2.3 Distributed Recognizer Management

A distributed recognition system may support a number of different recognizers that are controlled by a distributed recognition management system. These recognizers can include systems capable of supporting different classes of recognition, such as different languages, dialects, or accents, or cursive or boxed input for handwriting systems. When an application requires a recognition task to be performed, it first queries the recognition manager to find a recognizer that matches the parameters of the input to be recognized (as depicted in Figure 7). The recognition manager then queries each recognizer to find a system that supports the parameters specified by the application. When a recognizer indicates support for the specified parameter set, the enumeration ends and the selected recognizer is passed to the application. Note that the individual recognizers do not need to be centralized and may be distributed throughout the system, since the recognition manager acts as a controller for the set of recognizers.

2.4 User-Specific Dictionaries

Distributed recognition systems can also support user dictionaries, which are user-specific word lists (and possibly associated a-priori probabilities) that include words that a user writes frequently but which are unlikely to appear in a standard dictionary (examples include company names, work or personal interest specific terms, etc.). User dictionaries can be stored and managed centrally so that words added to the dictionary when using one application are available to all applications for context processing. Obviously, applications can manage and use their own local user-specific dictionaries if required, since they have full control over context decoding.

When an application requires the recognition of a signal that may contain words found in the user dictionary (e.g. standard handwritten text input such as the subject line of an e-mail or an arbitrary voice message), the centralized recognition system generates the usual intermediate recognition results to be returned to the application for context decoding. However, in addition to this it decodes the intermediate results using the user-dictionary as a language model, the result of which is also returned to the application. These two intermediate results structures can be combined by the application during its context decoding to generate a final decoding that includes the user-specific dictionary information.

2.5 User-Specific Training

Distributed recognition systems may also support user-specific training for recognizers, as depicted in Figure 8. The data generated by a user-specific recognition training application is submitted to the centralized recognition manager, which stores the data in a database. The recognition manager then enumerates all recognizers to determine if they support the data format as defined by the parameters associated with the training data, and if so, submits the training data to the recognizer for user-specific training.

When an existing recognizer is upgraded or a new recognizer is added to the system, the recognition manager queries the training database to determine if any training data of the format required by the recognizer exists. If so, the training data is submitted to the newly registered recognizer for processing, as depicted in Figure 9.

3 Conclusion

A number of techniques that allow pattern recognition to be performed in a distributed system have been discussed, including a method of returning the intermediate results generated by a pattern classifier to an application for context processing. In addition to this, techniques for managing multiple recognizers, user-specific dictionaries, and the user-specific training of recognizers have been given.

4 References

[1] Silverbrook Research, Netpage System Design Description, 8 September 2000.

[2] ParaGraph, Handwriting Recognition for Internet Connected Device, November 1999.

[3] Philips, SpeechMagic 4.0, 2000.

[4] L. Schomaker, H. Teulings, E. Helsper, and G. Abbink, "Adaptive Recognition Of Online, Cursive Handwriting", Proceedings of the Sixth International Conference on Handwriting and Drawing. Paris, July, 4-7 Telecom, (pp. 19-21), 1993.

[5] S. Connell and A.K. Jain, "Writer Adaptation of Online Handwritten Models," Proc. 5th International Conference on Document Analysis and Recognition, Bangalore, India, pp. 434-437, Sept. 1999.

[6] L. Rabiner and B. Juang, "Fundamentals of Speech Recognition", Prentice Hall, Englewood Cliffs, NJ, 1993.

[7] G. Lorette, "Handwriting Recognition or Reading? Situation At The Dawn of the 3rd Millennium", Advances In Handwriting Recognition, Series in Machine Perception and Artificial Intelligence, Vol. 34, pp. 3-15, World Scientific Publishing Co. 1999.

[8] H. Beigi and T. Fujisaki, "A Character Level Predictive Language Model and Its Application to Handwriting Recognition", Proceedings of the Canadian Conference on Electrical and Computer Engineering, Toronto, Canada, Sep. 13-16, 1992

[9] U. Marti and H.Bunke, "Handwritten Sentence Recognition", Proceedings of the 15th International Conference on Pattern Recognition, Barcelona, Spain, Volume 3, pp. 467-470, 2000.

[10] D. Bouchaffra, V. Govindaraju, and S. Srihari, "Postprocessing of Recognized Strings Using Nonstationary Markovian Models", IEEE Transactions Pattern Analysis and Machine Intelligence, 21(10), pp. 990-999, October 1999.

[11] J. Pitrelli and E. Ratzlaff, "Quantifying the Contribution of Language Modeling to Writer-Independent On-line Handwriting Recognition", Proceedings of the Seventh International Workshop on Frontiers in Handwriting Recognition, Amsterdam, September 11-13 2000.

[12] R. Srihari, "Use of Lexical and Syntactic Techniques in Recognizing Handwritten Text", ARPA Workshop on Human Language Technology, Princeton, NJ, March 1994.

[13] L. Yaeger, B. Webb, and R. Lyon, "Combining Neural Networks and Context-Driven Search for On-Line, Printed Handwriting Recognition in the Newton", AI Magazine, Volume 19, No. 1, pp. 73-89, AAAI 1998.

[14] A. C. Camproux, P. Tuffery, S. Hazout, "Hidden Markov Model Approach For Identifying The Modular Framework Of The Protein Backbone", Protein engineering, 12(12), pp. 1063, December 1999.

[15] Y. He, A. Kundu, "2-D Shape Classification Using Hidden Markov Model", IEEE Transactions on Pattern Analysis, 13(11), November 1991.

[16] T. Ryden, T. Terasvirta, S. Asbrink, "Stylized Facts of Daily Return Series and the Hidden Markov Model", Journal of Applied Econometrics, 13(3), pp. 217, May 1998.

5 Figures

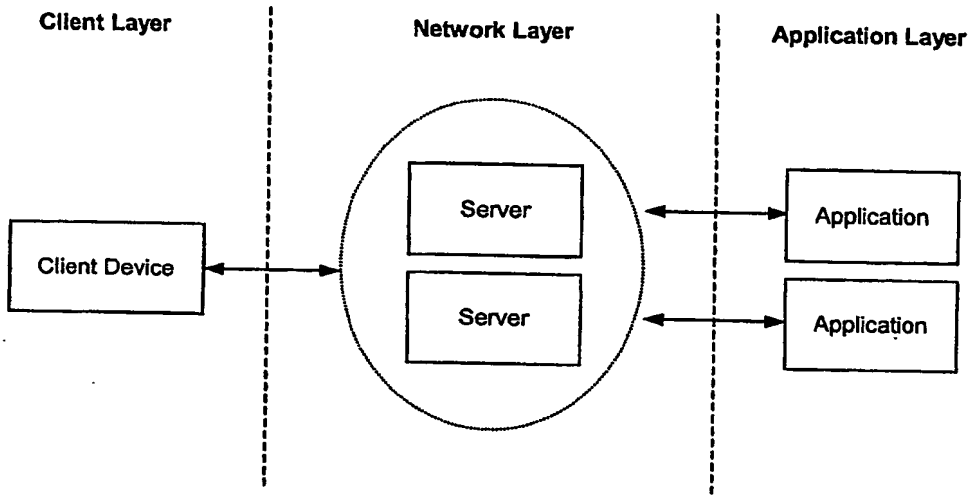

Figure 1. Distributed System Architecture

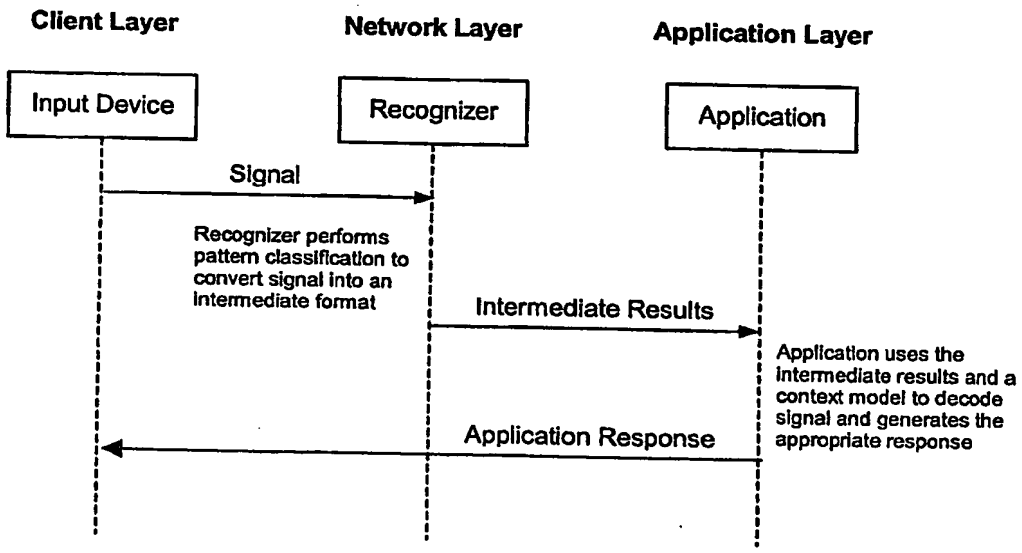

Figure 2. Distributed Recognition

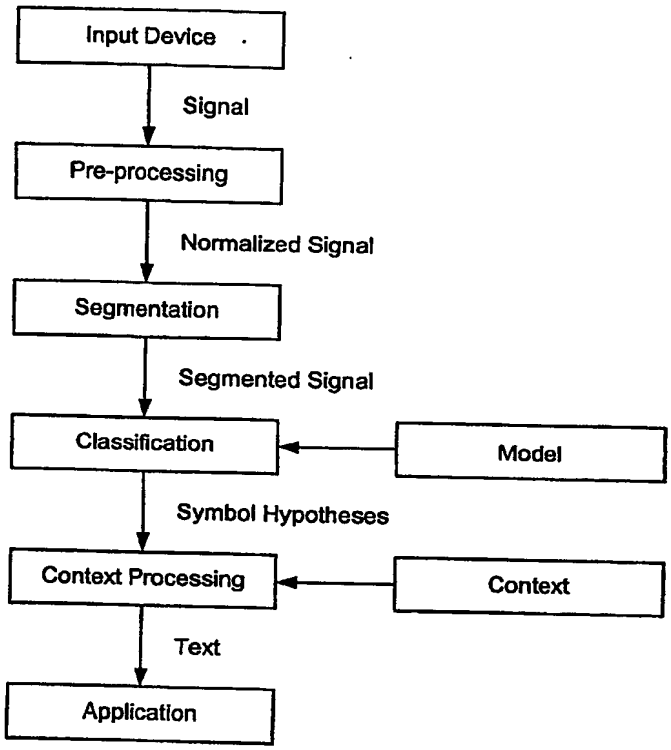

Figure 3. Basic Pattern Recognition

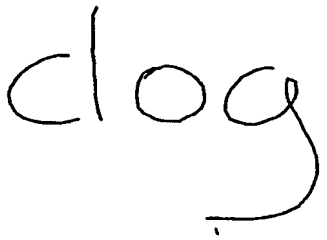

Figure 4. Ambiguous Input Ink for "clog"/"dog"

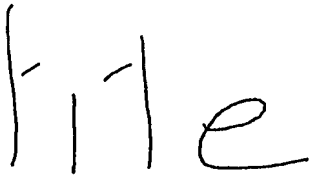

Figure 5. Ambiguous Input Ink for "tile"/"lite"

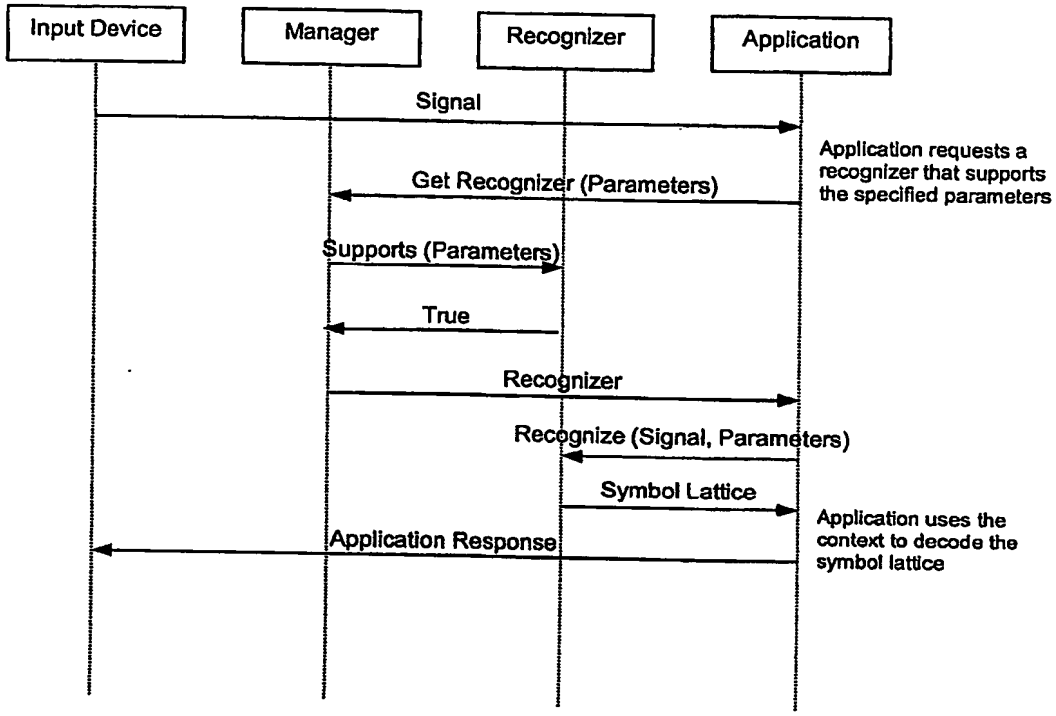

Figure 6. Example Recognition Scenario

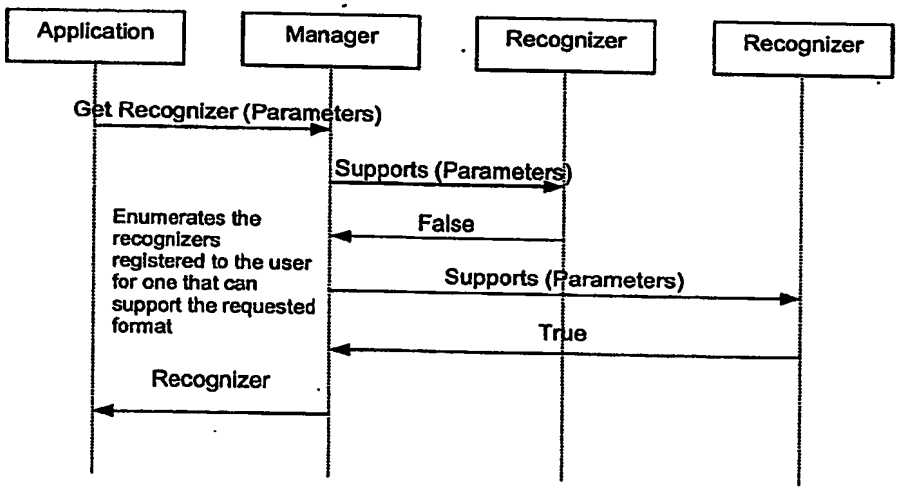

Figure 7. Recognizer Selection Scenario

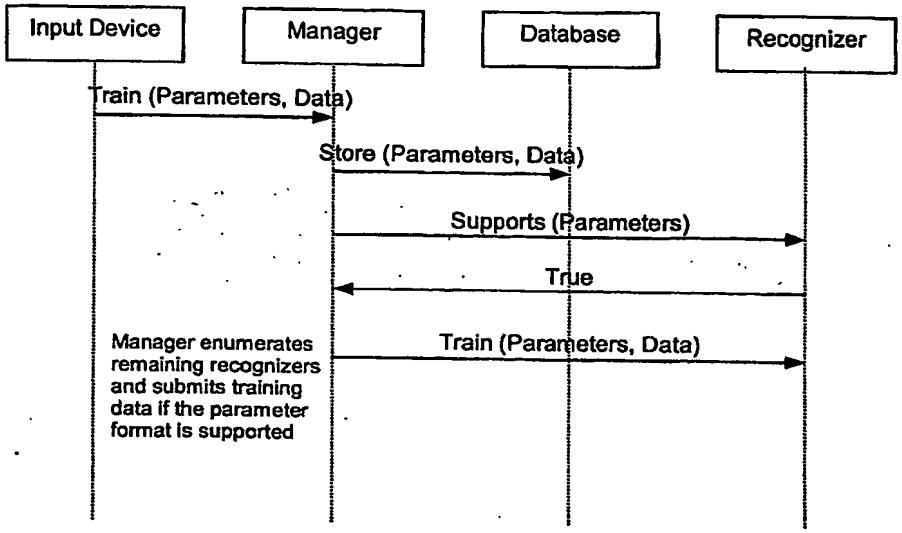

Figure 8. Recognizer Training Scenario

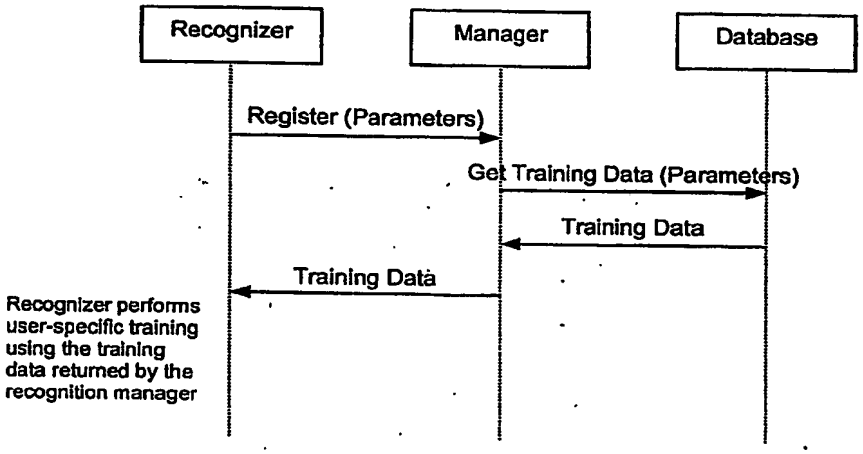

Figure 9. Recognizer Registration Scenario